Multi-Level Compliance Measurements for Software Process Appraisal

Suppasit Roongsangjan, Thanwadee Sunetnanta, Pattanasak Mongkolwat
Faculty of Information and Communication Technology, Mahidol University,
999 Phuttamonthon 4 Road, Salaya, Nakhon Pathom 73170, THAILAND
suppasit.run@student.mahidol.ac.th, {thanwadee.sun, pattanasak.mon}@mahidol.ac.th

***Abstract*— Software process appraisal is to assess whether an implemented software process complies with a process reference model. To conduct the appraisal, the appraisal team will request an organization to provide objective evidence reflecting practice implementation. Then such evidence will be examined, verified, and validated to generate appraisal results. This evidence collection process is done after a process is implemented. To better prepare for software process appraisal, we argued that the compliance of a process can be measured prior to its implementation. In light of that, we proposed multi-level compliance measurements to determine process reference model compliance, in terms of *Process Model Readiness Score, Process Enactment Score, and Process Implementation Readiness Score*. These measurements help provide an insight analysis of where the problems of practice implementation lie, i.e. at process modeling, at process enactment, or at process implementation.**

Keywords—Software Process Appraisal, Software Process Improvement (SPI), Insight Analysis, Compliance Measurement, Process Reference Model (PRM), Process Enactment

I. INTRODUCTION

In a software development organization, organizational maturity can be measured by an appraisal process [1]. A software process appraisal determines strengths and weaknesses of an implemented software development process against a process reference model (PRM) by an appraisal team [2]. PRM is a collection of practices. Well-known examples of PRMs are ISO/IEC 12207 Systems and software engineering - Software life cycle processes [3], ISO/IEC 29110 Software engineering - Lifecycle profiles for Very Small Entities (VSEs) [4], and Capability Maturity Model Integration (CMMI) [2].

Objective evidence is a result of a physical implementation of a process model. It can be output work products and outcomes. During an appraisal process, an appraisal team analyzes appraisal requirements, develops an appraisal plan, and obtains and inventories initial objective evidence [5]. In so doing, the appraisal team members (ATMs) collect output work products and usually conduct interview sessions to obtain affirmations of the outcomes. ATMs use objective evidence to indicate PRM compliance. Typically, such an appraisal process is done after the process implementation has finished. Accordingly, PRM compliance, therefore, is a measure of the implementation of a process model.

To better prepare for software process appraisal, we argued that the compliance of a process can be measured prior to its implementation. That is, we can check model practice compliance from how the process is defined, i.e., its process model. In light of that, we proposed multi-level compliance measurements for software process appraisal. The measurements quantify the compliance in terms of *Process Model Readiness Score*, *Process Enactment Score*, and *Process Implementation Readiness Score* at process modeling, process enactment, and process implementation, respectively.

In the next section, we describe existing research works related to process compliance measurement and highlight our contribution in comparison with them. In Section III, we explain the proposed measurements. Section IV shows calculations of these measurements and their applications for insight analysis in process design, process implementation, and appraisal context. Section V concludes this paper.

II. PROCESS COMPLIANCE MEASUREMENTS AND RELATED WORKS

The recent study related to obstacles in software process improvement (SPI) from Khan et al. (2017) [6] discussed that the needs for process deployment techniques are more crucial than the needs for new SPI models. Process deployment is concerned with introducing and supporting a new process model in a working environment [7]. The effectiveness of a deployed process can be measured by using process enactment tools, such as Spider-PE [8], SysProVal [9], and Taba Workstation [10]. Spider-PE is a process enactment tool that shows process adherence to a PRM. SysProVal and Taba Workstation measure team performance by using time and effort. Spider-PI [11] is a process improvement tool that shows strengths, weaknesses, opportunities, and threats of a process model when compared with a PRM.

Some process enactment tools measure compliance of process model implementation against a process model. *Process enactment deviation* represents the difference between the implementation of a process model and the model itself. Full compliance means no deviation. Several types of process enactment deviations are listed in the work

of Thompson et al. (2007) [12]. The works of Smatti et al. (2015) [13] and Silva et al. (2011) [14] measured deviation levels by using criteria and predefined rules. They counted a number of unmet criteria as process enactment deviation measurement. He et al. (2009) [15] used process pattern to measure process enactment deviation, such as sequence, parallel, and choice of activities. The absent, skipped, or reverse order of the implemented tasks represent a non-compliance process model. The work of Huo et al. (2006a, b) [16], [17] and Hug et al. (2012) [18] used data mining techniques to find deviated process enactment.

While process enactment deviation is measured during process implementation, the compliance of process model implementation against a PRM is measured as part of software process appraisal. Software process appraisal tools usually follow the measurement framework defined by process assessment models, such as ISO/IEC 15504 Information technology - Software process assessment [19], and CMMI [2]. Examples of software process appraisal tools are SEAL QQ [20], Generic Software Process Assessment [21], Appraisal Assistant [22], Appraisal Wizard [23], SPiCE 1-2-1 [24], ProEvaluator [25], SelfVation [26], CERTICSys [27], and Assessment Visualisation Tool (AVT) [28]. The compliance of process model implementation against a PRM is usually represented in terms of process maturity and capability levels.

Unlike the existing works that we have reviewed, we proposed an additional level of compliance measurement between a process model and a PRM. This measurement represents the similarity of a process model to a PRM in an appraisal context. The work of Gerke et al. (2009) [29] also raised the importance of this measurement. The compliance distance from a process model to a PRM will help determine SPI effort needed to achieve maturity levels and capability levels defined by the PRM. Such compliance distance can be measured before we actually implement the process model, and thus encourage us to achieve the compliance by the design of process model. In the next section, we will explain how to combine this additional compliance measurement to the existing ones to create our model of multi-level compliance measurements for software process appraisal.

III. Multi-level Compliance Measurements for Software Process Appraisal

Fig. 1 illustrates our model of multi-level compliance measurements for software process appraisal. It is the subsequence of our work in [30] that focuses on the tasks in a process model. We argue that compliance measurement should not be limited only to be during an appraisal but it should be done from the design of process model to process model implementation, then to process appraisal. Therefore, we propose three levels of compliance measurements which suit different stages of work towards software process appraisal. They are *Process Model Readiness Score*, *Process Enactment Score*, and *Process Implementation Readiness Score*. Firstly, *Process Model Readiness Score* measures the compliance between a PRM and a process model to reflect compliance by design of the process model. Secondly, *Process Enactment Score* measures the compliance between the deployed process model and the process model itself to reflect compliance by the enactment of the process model. Finally, *Process Implementation Readiness Score* measures the compliance between the deployed process model and the PRM to reflect compliance by the implementation of the process model.

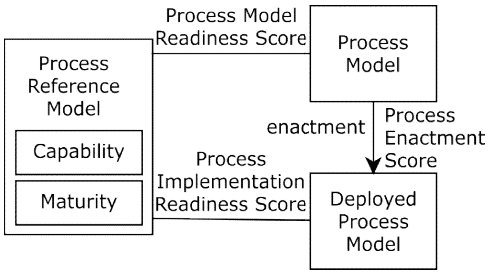

Figure 1 Multi-Level Compliance Measurements

Note that our model does not attempt to measure capability and maturity levels as the ones defined by CMMI [2], and those of ISO/IEC 15504 [19]. Instead, each level of measurements in our model aims to quantify the degree of effort required to achieve those capability and maturity levels. The following subsections will describe how these measurements are calculated.

A. Process Model Readiness Score

As mentioned earlier, the *Process Model Readiness Score* represents compliance by design. Typically, a PRM consists of a collection of practices that define good, common activities. These practices are required to be implemented in a process model. To put it differently, a PRM defines the required practices. A process model defines the implemented practices. A process in ISO/IEC 15504 is used for grouping practices of the same activity in the same way as a process area in CMMI does. The implementation of these practices in a process model represents process model readiness for the assessment by following the particular PRM.

This work defines two means to represent this measurement. The *c-score* is a ratio of the implemented practices of a process model to the required practices of the process area of a PRM. It represents the degree of achievement of process capability. The *m-score* is a ratio of the implemented practices of a process model to the required practices of a whole PRM. It represents the degree of achievement of process maturity. A process area a has practices in a PRM, or $a \subset PRM$, where PRM represents a set of practices in a PRM. We write $c^{p}_{PRM(a)}$ for the *c-score* of a process model p for the process area a in the PRM. A pair of vertical bars around a set name means a number of elements of that set. We define $Implemented\ Practices^{p}_{PRM(a)}$ for the set of practices in a that is implemented in p and define $Practices_{PRM(a)}$ for the set of practices in a. The $c^{p}_{PRM(a)}$ can be calculated by using the following equation:

$$c^{p}_{PRM(a)} = \frac{|Implemented\ Practices^{p}_{PRM(a)}|}{|Practices_{PRM(a)}|}$$

Since process area is a focused group of practices in a PRM for a certain activity, the *c-score* is also the measurement for the particular process, such as requirements, planning, or configuration management. It is useful for practitioners to use this score as an improvement indicator for the concerning software development activity.

We define the *m-score* for process model readiness for a whole PRM. It is a ratio of the implemented practices of a process model to the required practices of a PRM. We write m^{p}_{PRM} for the *m-score* of a process model *p* for a *PRM*. It can be calculated by using the following equation:

$$m^{p}_{PRM} = \frac{|Implemented\ Practices^{p}_{PRM}|}{|Practices_{PRM}|}$$

This measurement represents the degree of the required practices implemented in a process model. However, a group of all practices in a PRM represents the highest maturity level of that PRM. Maturity level 5 is the highest maturity level of CMMI and ISO/IEC 15504. It means that $m^{p}_{CMMI(5)}$ equals m^{p}_{CMMI} and $m^{p}_{ISO/IEC\ 15504(5)}$ equals $m^{p}_{ISO/IEC\ 15504}$, where *CMMI(5)* represent maturity level 5 of CMMI and *ISO/IEC 15504(5)* represent maturity level 5 of ISO/IEC 15504.

In case that an organization needs to measure a process model readiness with the lesser maturity level of a PRM, the calculation for the *m-score* must be applied for the smaller set of practices. We define $Practices_{PRM(l)}$ for the set of practices for maturity level *l* of a *PRM*. The *m-score* of the process model *p* for maturity level *l* of a *PRM* can be calculated by using the following equation:

$$m^{p}_{PRM(l)} = \frac{|Implemented\ Practices^{p}_{PRM(l)}|}{|Practices_{PRM(l)}|}$$

A process designer can use this score as an improvement indicator for the maturity level of the selected PRM. For example, an organization must implement every practice in seven process areas and ten generic practices to achieve the CMMI maturity level 2. If they aim to achieve CMMI maturity level 3, they must implement the additional practices in eleven process areas and two additional generic practices.

Process capability focuses on predictable results in particular process performance objectives [31]. Process maturity measures process controllability in an organization [32]. CMMI uses generic practices to represent the gap between process capability and process maturity. It shows that the implementation of every practice in every process area does not represent the highest maturity level. Generic practice is applied to many process areas. For example, generic practice 2.2 *Plan the Process* is an activity that must be implemented in every process area, such as a plan for performing the requirements management process, a plan for risk management process, and a plan for process and product quality assurance process. These plans must be included in a project plan. Moreover, the project planning process itself must implement this generic practice as an activity to "plan the plan" [2].

We use the difference in the degree of achievement of process capability and the degree of achievement of process maturity to represent the degree of generic practices implementation in a process model. We focus on the highest maturity level, thus we use m^{p}_{PRM} for the degree of achievement of process maturity. The degree of achievement of process capability is the average of *c-score* of every process area. We use *n* to represent a number of process areas to calculate average *c-score*. The calculation is shown in the following equation:

$$\overline{c^{p}_{PRM}} = \left(\sum_{a \subset PRM} (c^{p}_{PRM(a)})\right)/n$$

The *c-score* is calculated from specific practices, but the *m-score* includes both specific practices and generic practices. In the same process model, the *c-score* is always larger or equal to the *m-score*. We define $m^{p}_{PRM(g)}$ for the degree of generic practices implementation in a process model. This measurement can be calculated by using the following equation:

$$m^{p}_{PRM(g)} = \overline{c^{p}_{PRM}} - m^{p}_{PRM}$$

A process designer can use this measurement to check the implementation of generic practices in a process model. For example, a process model that is good for each process area has the $\overline{c^{p}_{PRM}}$ as 1.00. If this process model does not implement generic practices at all, we assume that the m^{p}_{PRM} equals 0.90. The implementation degree of generic practices in the *PRM* in the process model *p*, or $m^{p}_{PRM(g)}$ is 1.00-0.90 = 0.10.

The ultimate goal of SPI initiative is a matured process that is the implementation of the process model with the highest maturity level. Such model has m^{p}_{PRM} equal to 1.00. In practice, when designing a process model, a process designer would implement more tasks that would complement this goal. We write $(m^{p}_{PRM})^{c}$ to represent this complement. It is the degree of effort required to reach the mature process model. The degree of effort required for the maturity level *l* of the *PRM* of the process model *p* can be calculated by using the following equation:

$$(m^{p}_{PRM(l)})^{c} = 1 - m^{p}_{PRM(l)}$$

This measurement can be applied for the particular process area to represent the degree of effort required to reach full capability of that process area. We write $(c^{p}_{PRM(a)})^{c}$ for this effort and it can be calculated by using the following equation:

$$(c^{p}_{PRM(a)})^{c} = 1 - c^{p}_{PRM(a)}$$

The ATMs and appraisal participants can benefit from the *Process Model Readiness Score*. The ATMs can determine process maturity of a process to be assessed during the readiness review after they get initial objective evidence. They can use this score to support an on-site visiting plan to collect more evidence. They can use the degree of effort required for the maturity level l, or $(m^p_{PRM(l)})^c$ to represent an implementation gap. This gap can be used to support the suggestions about the unimplemented activities. These suggestions represent process improvement opportunities in which the appraisal participants will benefit from the SPI initiative.

Appraisal participants, process designers, in particular, can use this score to evaluate a process model before deployment process. If they add some tasks that do not change this score, these tasks may focus on a different detail level or scope. For example, a task to elicit system level requirements and a task to elicit Use-Case level requirements is implemented to understand requirements. Both tasks are needed, although they do not increase this score.

B. Process Enactment Score

The *Process Enactment Score* represents compliance by the enactment of a process model. It measures the completeness of process model implementation. The concept of process enactment concentrates on the enacted tasks and their output work products (WPs). The fully enacted process model means each task is performed and all output WP is created. A deployed task is a task that software development team members performed in their work. A set of deployed tasks is a subset of or equal to a set of tasks in a process model. A created output WP is a WP that is existed in a project repository. A set of created output WPs is a subset of or equal to a set of output WPs in a process model. A number of the created output WPs of a process model p, or $|Created\ Output\ WPs^p|$, and a number of the output WPs of a process model p, or $|Output\ WPs^p|$, is counted on a per-task basis. We write e^p for the enactment score of process model p. This measurement can be calculated by using the following equation:

$$e^p = \frac{|Deployed\ Tasks^p| + |Created\ Output\ WPs^p|}{|Tasks^p| + |Output\ WPs^p|}$$

This score equals 1.00 in the fully enacted process model. In a defined process, a project manager can use this score to monitor and control how a software development team follows a deployed process model. He or she can use this score to manage the team commitment to process model compliance. In a managed process, a process model may not complete, not well-defined, or the team may not follow a process model. A project manager can use this score to support how he or she manages the process.

The ATMs also benefit from *Process Enactment Score*. They can use this score as stopping criteria for objective evidence collection iteration. If this score does not increase, the team may assume that the process was enacted at that degree. We write $(e^p)^c$ for the effort to achieve the fully enacted process. This effort can be calculated by using the following equation:

$$(e^p)^c = 1 - e^p$$

This measurement represents a process enactment gap. This gap may exist through the fully matured process because SPI keeps on evolving a software development process. Process designers and process implementers can monitor this gap to manage the continuously improving software process.

C. Process Implementation Readiness Score

The *Process Implementation Readiness Score* represents compliance by the implementation of process model against a PRM. It is the overview of the degree of compliance by design and compliance by enactment. The full enactment of the mature process model has the full *Process Implementation Readiness Score*, or this score equals 1.00. This score is semantically equivalent to the highest maturity level in CMMI and ISO/IEC 15504.

The score calculation has two parts, *Process Model Readiness Score* and *Process Enactment Score*. We define i^p_{PRM} for *Process Implementation Readiness Score* for process model p against a *PRM*. This measurement can be calculated by using the following equation:

$$i^p_{PRM} = m^p_{PRM}\, e^p$$

This measurement can be applied for the particular maturity level or process area. The equations for the *Process Implementation Readiness Score* for the maturity level *l* and this score for the process area *a* can be written as follows:

$$i^p_{PRM(l)} = m^p_{PRM(l)}\, e^p\ ,\ i^p_{PRM(a)} = c^p_{PRM(a)}\, e^p$$

The *Process Implementation Readiness Score* shows the overview compliance degree by using *Process Model Readiness Score* and *Process Enactment Score*. This measurement emphasizes the proposed concept about the compliance measurement in a process model level in addition to the compliance measurement in process implementation level. On one hand, this work used *Process Model Readiness Score* and *Process Enactment Score* to create *Process Implementation Readiness Score*. On the other hand, this work digests an appraisal result into process model level and process model implementation level.

These measurements are useful for appraisal participants for an appraisal preparation. They can use these measurements to sketch summarizing the SPI effort used and to be used to achieve their SPI goal. The separation of concern in the compliance by design and compliance by enactment would help a process designer and a project manager to distinguish the improvement effort for a process model and process model implementation. Therefore, the proposed compliance measurements in process design and process enactment aspects would help to detect SPI problems and speed up the improvement cycle at both design time and enactment time.

TABLE I. Practices Implementation of Software Development Tasks and Output Work Products

Activity	Task Number	Task	Output Work Product							
			Vision	Glossary	System-Wide Requirements	Use Case	Use-Case Model	Work Items List	Iteration Plan	Test Case
Initiate Project	1*	Develop Technical Vision	SP 1.1, SP 1.2	SP 1.1, SP 1.2						
Plan and Manage Iteration	2	Plan Iteration						SP 1.2 +	SP 1.2 +	
Identify and Refine Requirements	3	Identify and Outline Requirements		SP 1.1, SP 1.2 +	SP 1.1, SP 1.2 +	SP 1.1, SP 1.2 +	SP 1.1, SP 1.2 +	SP 1.1, SP 1.2 +		
	4	Detail Use-Case Scenarios		SP 1.1 +		SP 1.1 +	SP 1.1 +			
	5	Detail System-Wide Requirements		SP 1.1 +	SP 1.1 +					
	6	Create Test Cases								+

'*' indicates an unimplemented task in process model enactment process.
'+' indicates output work product existence in a project repository.

IV. Applications of the Proposed Measurements

The proposed measurements involve three processes in the SPI cycle, which are measure, analyze, and change [32]. Process designing results as the changing of a process model. After the updated process model is implemented, a project manager and a process designer can use *Process Enactment Score* to monitor and analyze process model implementation to update the implemented process model in the next SPI cycle.

We use OpenUp process model from Eclipse Process Framework (EPF) [33] and CMMI [2] to demonstrate how the proposed measurements are used to improve the PRM compliance. OpenUp process model from EPF is a public process model resource that provides software development task description, purpose and output work products to be used in this work. This demonstration concentrates on Inception phase and the first specific goal (SG1 Manage Requirements) of the Requirements Management (REQM) process area of CMMI.

We use Table I to show some model elements in the OpenUp process model and the related specific practices in CMMI. This table shows three requirements-related activities in Inception phase in the OpenUp process model, *Initiate Project, Plan and Manage Iteration, Identified and Refine Requirements*, and *Agree on Technical Approach* activities. Activity is broken down into tasks. A task is the process element that we can assign a unit of work to be performed by roles [34]. This table has six tasks. The first task is *Develop Technical Vision* task in *Initiate Project* activity. This task has two output work products, *Vision,* and *Glossary*. The second to the sixth task also has output work products. The implemented task creates output work products in a project repository. We use an asterisk symbol to indicate an unimplemented task, which does not create output work product. We use a plus symbol to indicate output work product existence in a project repository. These output work products are used to support practice implementation.

This work uses task description and purpose to determine whether the task indicates practice implementation or not. The process to identify the relationship between software development tasks in a process model and practices in a PRM is described in [30]. Table I show this relationship by placing practice number in this table to show the relationship between practice, task, and output work product. For example, *Develop Technical Vision* task indicates SP 1.1 and SP 1.2. An ATM will use *Vision* and *Glossary* to support the implementation of these two practices.

Table I also shows example practices in CMMI. SG 1 of REQM process area of CMMI has five specific practices (SP). SP 1.1 *Understand Requirements* practice is satisfied by one of the implementations of *Develop Technical Vision, Identify and Outline Requirements, Detail Use-Case Scenarios*, or *Detail System-Wide Requirements* tasks, or task number 1, 3, 4, and 5. An ATM will find *Vision, Glossary, System-Wide Requirements, Use Case, Use-Case Model, and Work Items List* to support SP 1.1 practice implementation in this example process model.

SP 1.2 *Obtain Commitment to Requirements* practice is satisfied by one of the implementations of *Develop Technical Vision, Plan Iteration*, or *Identify and Outline Requirements* tasks, or task number 1, 2, and 3. An ATM will find *Vision, Glossary, System-Wide Requirements, Use Case, Use-Case Model, Work Items List,* and *Iteration Plan* to support this practice implementation. SP 1.3 *Manage Requirements Changes* practice, SP 1.4 *Maintain Bidirectional Traceability of Requirements* practice, and SP 1.5 *Ensure Alignment Between Project Work and Requirements* practice are not satisfied. Thus, they do not show in this table.

The following subsections show the calculation of *Process Model Readiness Score, Process Enactment Score,* and *Process Implementation Readiness Score*. At the end of this section shows how to use these measurements for insight analysis.

A. Process Model Readiness Score

The measurement for *Process Model Readiness Score* of the example OpenUp process model against CMMI, or m_{CMMI}^{OpenUp} requires the complete set of software development tasks and CMMI practices to calculate, which requires more space. However, we show the *c-score* for REQM process area, or $c_{CMMI(REQM)}^{OpenUp}$, which presents the same calculation in the smaller scope. Table I shows two satisfied specific practices, SP 1.1 and SP 1.2. This process area has five practices, SP 1.1 to SP 1.5. Therefore, the process model capability score for the example OpenUp process model for Requirements Management (REQM) process area of CMMI, or $c_{CMMI(REQM)}^{OpenUp}$, is calculated as follows:

$$c_{CMMI(REQM)}^{OpenUp} = \frac{2}{5} = 0.40$$

The gap to reach full capability of this process area, or $(c_{CMMI(REQM)}^{OpenUp})^c$, is $1 - 0.40 = 0.60$.

B. Process Enactment Score

There are six tasks in Table I. The number of the tasks in this example, or $|Tasks^{OpenUp}|$, is 6. We assume that Task number 1 is not implemented in this example by using the asterisk symbol after the task number. This makes the number of the deployed tasks, or $|Deployed\ Tasks^{OpenUp}|$, equal to 6-1=5.

Task number 1 has two output work products. Task number 2 to task number 6 have 2, 5, 3, 2, and 1 output work products, respectively. The summation of the number of output work products of each task is 2+2+5+3+2+1=15. This number is the number of output work products in the example process model, or $|Output\ WPs^{OpenUp}|$. We assume that Task number 1 is not implemented in this example, then two output work products of this task are not included in the created output work products. The number

of the created output work products, or $|Created\ Output\ WPs^{OpenUp}|$, is 2+5+3+2+1= 13.

$$e^{OpenUp} = \frac{(5+13)}{(6+15)} = 0.86$$

C. Process Implementation Readiness Score

For the same reason in Subsection A, this work shows the calculation for the *Process Implementation Readiness Score* for REQM process area, or i_{REQM}^{OpenUp}. This measurement uses *Process Model Readiness Score* from Subsection A and *Process Enactment Score* from Subsection B to calculate the *Process Implementation Readiness Score* for REQM process area.

$$i_{CMMI(REQM)}^{OpenUp} = (0.40)(0.86) = 0.34$$

The *Process Model Readiness Score* for REQM process area $c_{CMMI(REQM)}^{OpenUp}$ shows that the example OpenUp process model does not achieve the capability of CMMI REQM process area. A process designer can drill down to the relationship between tasks and practices in Table I to identify that SP 1.3, SP 1.4, and SP 1.5 are not satisfied by the tasks in this example OpenUp process model.

Due to the CMMI maturity level determination rule, each practice in a goal must be satisfied. In this example, SP 1.3 to 1.5 are not satisfied; thus the first specific goal of the REQM process area is not reached and this example OpenUp process model does not achieve CMMI maturity level 2.

The *Process Model Readiness Score* shows at the very beginning in a process designing process that the *c-score* for this process model for REQM process area is less than 1.00. A process designer can use this score to identify that the SPI problem is in a process model. If a process designer wants to achieve full capability of this process area, he or she must add some activities with all required activity concepts.

- SP 1.3 requires activity to manage requirements changes.
- SP 1.4 requires activity to trace requirements.
- SP 1.5 requires activity to trace requirements and to validate requirements.

This example shows that the *Process Enactment Score* e^{OpenUp} does not reach 1.00. It means that the process model does not fully enact. The SPI problem is in enactment process. This problem usually occurs in a fast-paced for high maturity level organization. A process designer may try to deploy a process model that is over capacity for change of a software development team. This measurement would help to adjust SPI speed, or it would measure capacity for change of a software development team. For example, a software development team cannot perform every task and cannot create every output work products in a process model. The *Process Enactment Score* will go below 1.00. A process designer and a project manager should work together to concentrate on the commitment to follow the process model.

The *Process Implementation Readiness Score* for REQM process area $i_{CMMI(REQM)}^{OpenUp}$ shows the big gap to achieve the full capability of REQM process area. A process designer and a project manager must fill this gap by adding some activities and monitoring the process model enactment process, respectively.

V. Conclusion, Discussions, and Future Works

This paper presents the multi-level compliance measurements for software process appraisal. These measurements do not replace the measurement of the existing appraisal models, such as CMMI, and ISO/IEC 15504. Actually, the proposed measurements can support the currently available compliance measurements for insight analysis for the better appraisal preparation for both ATMs and appraisal participants.

These measurements consist of the *Process Model Readiness Score, Process Enactment Score,* and *Process Implementation Readiness Score*. They represent compliance by design, compliance by the enactment, and compliance by the implementation, respectively. These measurements can detect problems in SPI cycle, which are the process changing in process design time, and process measurement in process enactment time. A process designer and a project manager can use these measurements to speed up SPI cycle toward the matured process by using the measurements that focus on the degree of compliance that refers to the maturity and capability of the selected PRM. The earlier SPI problems detection before the actual appraisal process could help the organization to prepare for an appraisal.

However, the measurements that are based on a process model cannot detect the alternative practice that is not defined in a process model. This problem can occur in an organization with process maturity level 2 because a process model may not exist, not be well prepared, or a software development team may not follow it. The output work products that do not follow a process model seem to be the evidence for the alternative practices. An ATM must affirm whether these practices are managed or not managed to determine the practice satisfaction for the practices in maturity level 2. This problem will not occur in an organization with process maturity level 3 or more because they have a defined process. The alternative practices have to be defined in a process model.

As a proof of concept, we are developing the appraisal assistant tool that implements the concept in [30] and it will implement the proposed measurements in this work. The planned evaluation of this work is to compare the benefits of using and not using this tool in terms of the measurements for insight analysis to support SPI initiative.

Acknowledgment

We would like to thank Dr. Chayakorn Piyabunditkul, Software Engineering Specialist of the National Science and Technology Development Agency of Thailand (NSTDA)

for kindly giving suggestions. This research project was partially supported by the Faculty of Information and Communication Technology, Mahidol University.

REFERENCES

1. ISO, "ISO/IEC/IEEE 24765 Systems and software engineering - Vocabulary," Geneva, CH, Dec. 2010.
2. SEI, "CMMI for Development, Version 1.3 (CMMI-DEV, V1.3) Improving processes for developing better products and services," Nov. 2010.
3. ISO, "ISO/IEC 12207 Systems and software engineering - Software life cycle processes," 2008.
4. ISO, "ISO/IEC 29110 Systems and software engineering - Lifecycle profiles for Very Small Entities (VSEs)," 2016.
5. SCAMPI Upgrade Team, "Standard CMMI Appraisal Method for Process Improvement (SCAMPI) A , Version 1.3: Method Definition Document," *Management*, no. March, p. 245, 2011.
6. A. A. Khan, J. Keung, M. Niazi, S. Hussain, and A. Ahmad, "Systematic Literature Review and Empirical Investigation of Barriers to Process Improvement in Global Software Development: Client– Vendor Perspective," *Inf. Softw. Technol.*, 2017.
7. Inform-IT, *Foundations of IT Service Management Based on ITIL V3*, 1st ed. Van Haren Publishing, 2007.
8. C. Portela and A. Vasconcelos, "Spider-PE: A Set of Support Tools to Software Process Enactment," *ICSEA 2014 Ninth Int. Conf. Softw. Eng. Adv.*, no. c, pp. 539–544, 2014.
9. I. Garcia, C. Pacheco, and J. Calvo-Manzano, "Using a web-based tool to define and implement software process improvement initiatives in a small industrial setting," *IET Softw.*, vol. 4, no. 4, p. 237, 2010.
10. A. I. F. Ferreira *et al.*, "Taba Workstation: Supporting Software Process Improvement Initiatives Based on Software Standards and Maturity Models," *Softw. Process Improv. 13th Eur. Conf. EuroSPI 2006, Joensuu, Finland, Oct. 11-13, 2006. Proc.*, pp. 207–218, 2006.
11. L. P. Mezzomo, S. Ronaldo, A. Marcos, and L. De Vasconcelos, "A Set of Support Tools to Software Process Appraisal and Improvement in Adherence to CMMI-DEV," *ICSEA 2016 Elev. Int. Conf. Softw. Eng. Adv.*, no. CMMI, pp. 263–271, 2016.
12. S. Thompson, T. Torabi, and P. Joshi, "A Framework to Detect Deviations During Process Enactment," in *6th IEEE/ACIS International Conference on Computer and Information Science (ICIS 2007)*, 2007, pp. 1066–1073.
13. M. Smatti, M. Oussalah, and M. A. Nacer, "A review of detecting and correcting deviations on software processes," in *2015 10th International Joint Conference on Software Technologies (ICSOFT)*, 2015, vol. 1, pp. 1–11.
14. M. A. A. da Silva, R. Bendraou, J. Robin, and X. Blanc, "Flexible Deviation Handling during Software Process Enactment," in *2011 IEEE 15th International Enterprise Distributed Object Computing Conference Workshops*, 2011, pp. 34–41.
15. X. He, J. Guo, Y. Wang, and Y. Guo, "An Automatic Compliance Checking Approach for Software Processes," in *2009 16th Asia-Pacific Software Engineering Conference*, 2009, pp. 467–474.
16. M. Huo, H. Zhang, and R. Jeffery, "An Exploratory Study of Process Enactment As Input to Software Process Improvement," in *Proceedings of the 2006 International Workshop on Software Quality*, 2006, pp. 39–44.
17. M. Huo, H. Zhang, and R. Jeffery, "A Systematic Approach to Process Enactment Analysis as Input to Software Process Improvement or Tailoring," in *2006 13th Asia Pacific Software Engineering Conference (APSEC'06)*, 2006, pp. 401–410.
18. C. Hug, R. Deneckère, and C. Salinesi, "Map-TBS: Map process enactment traces and analysis," in *2012 Sixth International Conference on Research Challenges in Information Science (RCIS)*, 2012, pp. 1–6.
19. ISO, "ISO/IEC 15504 Information technology - Process assessment," 2004.
20. R. H. Lok and A. J. Walker, "Automated tool support for an emerging international software process assessment standard," in *Proceedings of IEEE International Symposium on Software Engineering Standards*, 1997, pp. 25–35.
21. O. R. Yürüm, Ö. Ö. Top, and O. Demirörs, "Assessing Software Processes over a New Generic Software Process Assessment Tool," *Coll. Econ. Anal. Ann.*, 2016.
22. F. Liang, T. Rout, and A. Tuffley, "Appraisal Assistant Beta." [Online]. Available: https://www.sqi.griffith.edu.au/AppraisalAssistant/about.html.
23. Integrated System Diagnostics Incorporated, "Appraisal Wizard." [Online]. Available: http://isd-inc.com/tools.appraisalWizard/.
24. HM&S IT-Consulting GmbH, "SPiCE 1-2-1." [Online]. Available: http://www2.hms.org/cms/en/default.html.
25. J. Moura, C. Xavier, A. Marcos, and L. De Vasconcelos, "ProEvaluator : Uma Ferramenta para Avaliação de Processos de Software," no. June 2008, pp. 201–214, 2008.
26. I. Garcia, C. Pacheco, and D. Cruz, "Adopting an RIA-Based Tool for Supporting Assessment, Implementation and Learning in Software Process Improvement under the NMX-I-059/02-NYCE-2005 Standard in Small Software Enterprises," in *2010 Eighth ACIS International Conference on Software Engineering Research, Management and Applications*, 2010, pp. 29–35.
27. D. C. Silva, A. Raldi, T. Messias, A. M. Alves, and C. F. Salviano, "A Process Driven Software Platform to Full Support Process Assessment Method," in *2014 40th EUROMICRO Conference on Software Engineering and Advanced Applications*, 2014, pp. 135–136.
28. R. Hunter, G. Robinson, and I. Woodman, "Tool support for software process assessment and improvement," *Softw. Process Improv. Pract.*, vol. 3, pp. 213–223, 1997.
29. K. Gerke, J. Cardoso, and A. Claus, "Measuring the Compliance of Processes with Reference Models," in *On the Move to Meaningful Internet Systems: OTM 2009*, vol. 5870, R. Meersman, T. Dillon, and P. Herrero, Eds. Springer Berlin / Heidelberg, 2009, pp. 76–93.
30. S. Roongsangjan, T. Sunetnanta, and P. Mongkolwat, "Using FCA Implication to Determine the Compliance of Model Practice Implementation for Software Process," in *Proceedings of the ICMSS 2017*, 2017, pp. 64–70.
31. NIST/SEMATECH, "e-Handbook of Statistical Methods," 2012. [Online]. Available: http://www.itl.nist.gov/div898/handbook/.
32. I. Sommerville, *Software Engineering*, 9th ed. Harlow, England: Addison-Wesley, 2010.
33. "OpenUp Process in Eclipse Process Framework Project (EPF)." [Online]. Available: http://epf.eclipse.org/wikis/openup/.
34. OMG, "Software & Systems Process Engineering Meta-Model Specification V2.0," no. April. p. 236, 2008.

